Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Computer Science 10 (2012) 263 – 271

Procedia Computer Science

The 3rd International Conference on Ambient Systems, Networks and Technologies (ANT)

Predicting Trust from User Ratings

Nikolay Korovaiko[a], Alex Thomo[a]

[a] *University of Victoria, 3800 Finnerty Road, Victoria BC V8P 5C2 ,Canada*

Abstract

Trust relationships between users in various online communities are notoriously hard to model for computer scientists. It can be easily verified that trying to infer trust based on the social network alone is often inefficient. Therefore, the avenue we explore is applying Data Mining algorithms to unearth latent relationships and patterns from background data. In this paper, we focus on a case where the background data is user ratings for online product reviews. We consider as a testing ground a large dataset provided by Epinions.com that contains a trust network as well as user ratings for reviews on products from a wide range of categories. In order to predict trust we define and compute a critical set of features, which we show to be highly effective in providing the basis for trust predictions. Then, we show that state-of-the-art classifiers can do an impressive job in predicting trust based on our extracted features. For this, we employ a variety of measures to evaluate the classification based on these features. We show that by carefully collecting and synthesizing readily available background information, such as ratings for online reviews, one can accurately predict social links based on trust.

Keywords: Social Computing, Trust Prediction, Trust Inference, Epinions, Data Mining

1. Introduction

With the explosive growth in popularity of social networks and e-commerce systems users are constantly in interaction with each other. The trust factor plays an important role in initiating these interactions and building higher-quality relationships between the users. Even though trust takes many different meanings and highly depends on the context in which users interact with each other, it has been shown that trust can be approximated from other relationships.

Consider a few examples. In Epinions.com, trusting a particular person often means that the reviews written by that person are highly appreciated or that the person has preferences similar to the trustor. E-commerce systems suggest another example. We are more willing to buy an item from a particular seller on E-Bay or Amazon, if either we or our friends had positive experience with that seller in past. On the other hand, we are reluctant to engage in any relationship with strangers. On freelance websites trust means fruitful agreements between a professional and employer. Dating services might try to leverage users' preferences to help their users find a perfect match.

Our attitudes towards trust are often very different and individual. One might believe that a particular

Email addresses: nikolayk@uvic.ca (Nikolay Korovaiko), thomo@cs.uvic.ca (Alex Thomo)

1877-0509 © 2012 Published by Elsevier Ltd. Open access under CC BY-NC-ND license.
doi:10.1016/j.procs.2012.06.036

seller on E-Bay provides an excellent service, even though this seller sometimes delays shipping by a week. For another person any delay might be unacceptable. Trust-aware systems can help users make the right choices and have relationships that lead to positive outcomes.

1.1. The Trust Prediction Problem and Our Contributions

In online communities, users interact with each other in many ways. On Epinions for example (which is focus of this paper), the active users participate in discussions that grow around various products and write reviews on these products. The rest of the user community comments and rates the reviews. Additionaly, users can specify whom they trust. These trust connections constitute the user's trust network. The problem we study here is how to predict these trust links. This is an important problem which, when solved effectively, enhances the user online experience by connecting her to peers who share the same interests and values. The users can rely on the input from their peers or trustees to form their own opinion about a particular product much faster and easier. Incorporating this background information into a trust prediction algorithm delivers more accurate results in general. Furthermore, we might be able to infer trust relationships in cases where the traditional trust propagation algorithms fail. For instance, by using background information in the form of user ratings for online reviews we might be able to find users who have similar preferences and thus probably trust each other even though, in terms of the current trust graph, they appear to be quite far from each other.

There are various reasons why we decide to trust another person. We might know a person for a long time or we might share many interests in common. The person might be very reliable and trustworthy or just knowledgeable in a particular topic. There are a few key factors affecting the decision of a particular user to trust another one. First, both users might simply have very similar preferences. In other words, they tend to like the same items. Second, the user can trust the other one if she decides that the person is a good reviewer who writes high quality reviews on some products on Epinions. Third, the user might think of the other person as being a good review critic. Finally, both users might be friends. In the latter case, there is typically a mutual trust link between the users, even if they do not have that many things in common. We capture these ideas by computing a series of features from the data. Then, we use these features to build trust prediction models that significantly outperform state-of-the-art methods for trust prediction. In general, we achieve improvements of about 5-20% in performance (e.g. precision, recall, F-measure and Roc Area) over other competing approaches. We also apply various Data Mining techniques to our features in order to incorporate the biases from pairs of users (for whom we try to predict trust) and evaluate the impact that their biases have on performance.

2. Related Work

Jennifer Golbeck was one of the first pioneers to research the problem of trust prediction from a Computer Science perspective. In [1], Golbeck discusses various properties of trust such as *transitivity*, *composability* and *asymmetry*. Then, she proposes algorithms for inferring binary and continuous trust values from trust networks, based on trust propagation. Golbeck also suggests another trust inference algorithm called *Sunny* [2]. The algorithm uses a probabilistic sampling technique to estimate our confidence in the trust information from some designated sources. In [3], the authors develop a taxonomy of user relationships for the Epinions dataset. This taxonomy is used to obtain an extensive set of simple features which is in turn employed for training Naive Bayes and SVM classifiers. However, one should note that it is not always feasible to employ the overwhelmingly large number of features suggested by the authors. Moreover, some of features can be very naturally combined into a single one resulting in more accurate predictions. Viet-An Nguyen and *et. al* [4] derive several trust prediction models from a well-studied Trust Antecedent Framework used in management science. The framework captures the three following factors: ability, benevolence and integrity. The authors approximate each factor through a set of quantitative features used for training a SVM classifier. In another publication [5], various features from writer-reviewer interactions are derived and used in personalized and cluster-based classification methods. The former trains one classifier for each user using user-specific training data. The cluster-based method first constructs user clusters before training one classifier for each user cluster.

3. Features

Users interact, often implicitly, with each other in online communities, such as Epinions. In particular, in the Epinions case, users write reviews about different products as well as rate the reviews of other people. The users can also create a network of trusted users by issuing trust statements. Why do users issue such trust statements? Their main reason is to express their liking of the reviews written by the trusted user. Then, in the future, the users focus first on reading the reviews of the users they trust. However, users often need help in determining whom to trust. In order to help users in this discovery, an intelligent trust recommendation system needs to be build. This system would try to accurately predict trust among users. The predictions could be turned into effective trust recommendations that will greatly enhance the online experience of users.

We propose the following parameters to explore for predicting uni- or bi- directional trust (or distrust) between two users u and v.

1. u and v give similar ratings to the reviews they read
2. u and v are interested in similar categories of products
3. u and v produce reviews in the same categories that interest them
4. u and v rate the reviews produced by the same reviewers
5. u gives high ratings to reviews produced by v
6. u anonymizes a considerable number of ratings for reviews produced by v
7. v is a reputable reviewer
8. u and v have the same trustees.

Of course, the users might trust each other due to reasons other than the above. For example, they might have been friends for a long time. If this is the case, the users might not have many things in common, even though there are mutual trust links between them. Such trust links should be treated differently from regular ones or even filtered out, so that they do not introduce extra noise into the trust prediction algorithms.

In order to capture the above eight listed parameters in terms of formal features, we first introduce some notation. In the following we will interchangeably use item and review. Let u, v, y be users. We denote by

U	the set of users
I_u	the set of items rated by u
$I_{u,r}$	the set of items rated r (where $r \in \{1, 2, 3, 4, 5\}$) by u
$I_{u,c}$	the set of items in category c rated by u
$I_{u,y}$	the set of reviews (items) produced by y and rated by u
J_v	the set of reviews (items) produced by v
C_u	the set or multiset (depending on the feature) of categories of the items in I_u
D_u	the set or multiset (depending on the feature) of categories of the reviews (items) produced by u
Y_u	the set or multiset (depending on the feature) of reviewers (users) who have produced the reviews (items) in I_u
T_u	the set of trustees of user u, *i.e.* those users that u trusts.

For simplicity we will denote by $r_{u,i}$ a rating (u, i, r) that a user u gives for item i. This also reflects the fact that for a given user and a given item there can be not more than one rating.

We treat trust prediction as a classification problem, that is, for each ordered pair (u, v) of users, a new value called class (trust or distrust) has to be assigned. There is a rich repertoire of classifier algorithms available for the classification problem in the field. We choose to use Random Forests and Bayesian classifiers in certain cases. There are several options to convert the eight aforementioned parameters into a set of features.

Parameter 1: *u and v give similar ratings to the reviews they read.*

The first feature we propose is to represent both users as sets of their ratings and then compute the Pearson Correlation (PC) of these sets. Specifically, we define $f_{1,a}$ to be

$$f_{1,a} = \frac{\sum_{i \in I_u \cap I_v} (r_{ui} - \bar{r}_u)(r_{vi} - \bar{r}_v)}{\sqrt{\sum_{i \in I_u \cap I_v} (r_{ui} - \bar{r}_u)^2} \sqrt{\sum_{i \in I_u \cap I_v} (r_{vi} - \bar{r}_v)^2}}.$$

The second, third, fourth, and fifth attribute we propose are based on the number of partisan ratings, 5, 4, and 1, 2. Typically, ratings of 4 or 5 are a strong indicator of user likes, whereas ratings of 1 or 2 are a strong indicator of user dislikes. Intuitively, when u and v have a relatively significant number of compatible partisan ratings, their likes and dislikes are aligned. On the other hand, when u and v have incompatible partisan ratings, e.g. u gives a rating of 1 whereas v gives a rating of 5, their preferences exhibit a conflict.

The observations above can be converted into four features $f_{1,b}$, $f_{1,c}$, $f_{1,d}$, and $f_{1,e}$ we give below. These features are given in terms of Jaccard Similarity and they measure the relative weight of partisan agreements or disagreements. Let

$$I_{u,\uparrow} = I_{u,5} \cup I_{u,4}, \quad I_{u,\downarrow} = I_{u,1} \cup I_{u,2}.$$

Also similarly define $I_{v,\uparrow}$ and $I_{v,\downarrow}$. Now we define

$$f_{1,b} = \frac{|I_{u,\uparrow} \cap I_{v,\uparrow}|}{|I_{u,\uparrow} \cup I_{v,\uparrow}|}, \qquad f_{1,c} = \frac{|I_{u,\downarrow} \cap I_{v,\downarrow}|}{|I_{u,\downarrow} \cup I_{v,\downarrow}|}$$
$$f_{1,d} = \frac{|I_{u,\uparrow} \cap I_{v,\downarrow}|}{|I_{u,\uparrow} \cup I_{v,\downarrow}|}, \qquad f_{1,e} = \frac{|I_{u,\downarrow} \cap I_{v,\uparrow}|}{|I_{u,\downarrow} \cup I_{v,\uparrow}|}.$$

Parameter 2: *u and v are interested in similar categories of products.*

Often datasets, such as those from Epinions, do not contain explict user preferences for different item categories. However latent preferences can be discovered by considering the ratings the users have given to items belonging in given categories.

The first feature we define in this group is

$$f_{2,a} = \frac{|C_u \cap C_v|}{|C_u \cup C_v|}$$

which measures the amount of overlap between the categories of items that u and v have rated. Here C_u and C_v are considered to be multisets. We continue to use Jaccard Similarity in order to take into consideration not only the categories the users prefer in common, but also the users' range of activity.

Using this feature we are able to estimate the user similarity even in a case where users did not give a single rating to the same review. This is a very common scenario. For example, both users love to discuss sci-fi books. However, there are a large number of reviews on sci-fi literature, so users rarely rated the same ones. The feature appears to be less accurate, as it deals with aggregations by the categories rather than user ratings.

If a user is interested in a particular category, he typically reads and rates more items in that category. Counting the number of items for the category allows us to estimate the user interest in it. Next, we compute the Pearson Correlation for the two sets (of u and v) of those counts. Formally, we have

$$f_{2,b} = \frac{\sum_{c \in C_u \cap C_v} \left(|I_{u,c}| - \frac{|I_u|}{|C_u|}\right)\left(|I_{v,c}| - \frac{|I_v|}{|C_v|}\right)}{\sqrt{\sum_{c \in C_u \cap C_v} \left(|I_{u,c}| - \frac{|I_u|}{|C_u|}\right)^2} \sqrt{\sum_{c \in C_u \cap C_v} \left(|I_{v,c}| - \frac{|I_v|}{|C_v|}\right)^2}}.$$

In this definition, C_u and C_v are considered as sets.

Another way to receive the estimate of the user's interest in different categories is to compute the user's average ratings for each of the categories. Then we compute the Pearson Correlation for the sets of these

averages for u and v.

$$f_{2,c} = \frac{\sum_{c\in C_u\cap C_v}(r_{u,c}-\hat{r}_u)(|r_{v,c}|-\hat{r}_v)}{\sqrt{\sum_{c\in C_u\cap C_v}(r_{u,c}-\hat{r}_u)^2}\sqrt{\sum_{c\in C_u\cap C_v}(r_{v,c}-\hat{r}_v)^2}}$$

where

$$r_{u,c}=\frac{\sum_{i\in I_{u,c}} r_{u,i}}{|I_{u,c}|}, \qquad \hat{r}_u=\frac{\sum_{c\in C_u} r_{u,c}}{|C_u|}$$
$$r_{v,c}=\frac{\sum_{i\in I_{v,c}} r_{v,i}}{|I_{v,c}|}, \qquad \hat{r}_v=\frac{\sum_{c\in C_v} r_{v,c}}{|C_v|}.$$

For this definition as well, C_u and C_v are considered as sets.

Parameter 3: *u and v produce reviews in the same categories that interest them.*

For this parameter we employ the Jaccard Similarity with respect to the categories of the reviews user u and v have produced. Specifically, we propose the following feature.

$$f_3 = \frac{|D_u\cap D_v|}{|D_u\cup D_v|}.$$

In this definition, D_u and D_v are considered as multisets.

Parameter 4: *u and v rate reviews produced by the same reviewers.*

Another indication of similar user preferences is when both users favor the reviews written by the same reviewers. Typically, if a user likes a reviewer, the user gives higher ratings to reviews from that reviewer. It might seem that this parameter is always correlated with the first one, as the same rating information is used. However, there is often no correlation between two. Consider an example. User u rates the review i written by reviewer r. User v rates a *different* review, j, produced by the same reviewer r. The occurrences of this case are very frequent.

As this feature deals with the rating information aggregated by the reviewers, it again alleviates the sparseness of the ratings. Due to the fact that the majority of the reviewers focus on very specific topics which span several categories or, conversely, some partition of a particular category, the feature is potentially as accurate as the one based on the user ratings.

We assume that the average rating given by a user to the reviews from a reviewer reflects the user's preferences towards the reviewer. We employ the Pearson correlation to compute the similarity between the two sets of average ratings given by u and v to reviewers. Formally,

$$f_4 = \frac{\sum_{y\in Y_u\cap Y_v}\left(r_{u,y}-\check{r}_u\right)\left(|r_{v,y}|-\check{r}_v\right)}{\sqrt{\sum_{y\in Y_u\cap Y_v}\left(r_{u,y}-\check{r}_u\right)^2}\sqrt{\sum_{y\in Y_u\cap Y_v}\left(r_{v,y}-\check{r}_v\right)^2}}$$

where

$$r_{u,y}=\frac{\sum_{i\in I_{u,y}} r_{u,i}}{|I_{u,y}|}, \qquad \check{r}_u=\frac{\sum_{y\in Y_u} r_{u,y}}{|Y_u|}$$
$$r_{v,y}=\frac{\sum_{i\in I_{v,y}} r_{v,i}}{|I_{v,y}|}, \qquad \check{r}_v=\frac{\sum_{y\in Y_v} r_{v,y}}{|Y_v|}.$$

In this definition, Y_u and Y_v are considered as sets.

Parameter 5: *u gives high ratings to reviews produced by v.*

An approach is to compute an average rating that the user gives to the reviews (items) produced by v. However, the baseline predictors technique suggested in [6]

allows us to improve on this crude average by including the linear sum of four components. The first component is the global average of all ratings in our sampled dataset, which we denote by $\bar{r}$. The second, third, and fourth components are the differences from the global average of the following averages:

- the average of all ratings given to the items produced by v, denoted by $\check{r}_v$
- the average of all ratings u gives, denoted by $\bar{r}_u$
- the average of all ratings that u gave to the items produced by v, denoted–similarly as for Parameter 4–by $r_{u,v}$.

We have

$$f_{6,a} \quad = \quad \bar{r} + (\check{r}_v - \bar{r}) + (\bar{r}_u - \bar{r}) + (r_{u,v} - \bar{r})\,.$$

Two other features we define are the fraction of high (low) ratings u gives to reviews (items) produced by v. Namely, we have

$$f_{6,b} = \frac{|I_{u,\uparrow} \cap J_v|}{|I_{u,\uparrow}|}, \text{and} \quad f_{6,c} = \frac{|I_{u,\downarrow} \cap J_v|}{|I_{u,\downarrow}|}.$$

Parameter 6: *u anonymizes a considerable number of ratings for reviews produced by v.*

We start by computing the ratio of anonymized ratings u gives to the v's items, $\frac{|I^-_{u,v}|}{|I_{u,v}|}$, where $I^-_{u,v} \subseteq I_{u,v}$ is the set of v's items rated anonymously by u.

We then we consider the high or low ratings only, and have $\frac{|I^-_{u,v,\uparrow}|}{|I_{u,v,\uparrow}|}$ ($f_{6,d}$) and $\frac{|I^-_{u,v,\downarrow}|}{|I_{u,v,\downarrow}|}$ ($f_{6,e}$), where $I^-_{u,v,\uparrow}$, $I_{u,v,\uparrow}$, $I^-_{u,v,\downarrow}$, and $I_{u,v,\downarrow}$ are defined as their non-arrow counterparts, but considering the high or low ratings only.

The baseline predictors technique can be also applied to these ratios. For this, let

R the set of all ratings

R^- the set of all anonymous ratings ($R^- \subseteq R$)

$R_{\rightarrow v}$ the set of all ratings for v's items

$R^-_{\rightarrow v}$ the set of all anonymous ratings for v's items

$R_{u\rightarrow}$ the set of all u's ratings

$R^-_{u\rightarrow}$ the set of all anonymous u's ratings.

Also let $R_\uparrow$, $R^-_\uparrow$, $R_{\rightarrow v,\uparrow}$, $R^-_{\rightarrow v,\uparrow}$, $R_{u\rightarrow,\uparrow}$, $R^-_{u\rightarrow,\uparrow}$ be defined similarly as above, but with only the high ratings considered. Likewise for $R_\uparrow$, $R^-_\downarrow$, $R_{\rightarrow v,\downarrow}$, $R^-_{\rightarrow v,\downarrow}$, $R_{u\rightarrow,\downarrow}$, $R^-_{u\rightarrow,\downarrow}$ but with only the low ratings considered.

We now define

$$f_{6,a} = \frac{|R^-|}{|R|} + \frac{|R^-_{\rightarrow v}|}{|R_{\rightarrow v}|} + \frac{|R^-_{u\rightarrow}|}{|R_{u\rightarrow}|} + \frac{|I^-_{u,v}|}{|I_{u,v}|}, \qquad f_{6,b} = \frac{|R^-_\uparrow|}{|R_\uparrow|} + \frac{|R^-_{\rightarrow v,\uparrow}|}{|R_{\rightarrow v,\uparrow}|} + \frac{|R^-_{u\rightarrow,\uparrow}|}{|R_{u\rightarrow,\uparrow}|} + \frac{|I^-_{u,v,\uparrow}|}{|I_{u,v,\uparrow}|}$$

$$f_{6,c} \quad = \quad \frac{|R^-_\downarrow|}{|R_\downarrow|} + \frac{|R^-_{\rightarrow v,\downarrow}|}{|R_{\rightarrow v,\downarrow}|} + \frac{|R^-_{u\rightarrow,\downarrow}|}{|R_{u\rightarrow,\downarrow}|} + \frac{|I^-_{u,v,\downarrow}|}{|I_{u,v,\downarrow}|}.$$

In $f_{6,a}$ the first component is how often, on average, users decide to keep their ratings anonymous, the second is how often, on average, v receives anonymous ratings, and the third is how often, on average, u gives anonymous ratings. Similar comments can also be made for the $\uparrow$ and $\downarrow$ components, but considering the high or low ratings only.

Parameter 7: *v is a reputable reviewer and u is a lenient rater.*

This parameter deals with a reviewer reputation and how the reputation affects the readers' decision to trust the reviewer. Typically, the users express their appreciation to the reviewer by giving higher ratings to his items. The more positive ratings a reviewer receives, the higher his reputation. This observation can be

converted into a feature by computing the difference of the average rating given to the items produced by v from the overall average rating in the dataset.

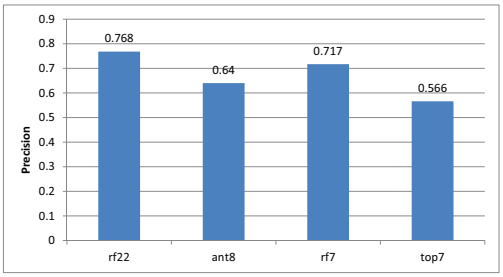
$$f_{7,a} = \check{r}_v - \bar{r}.$$

On the other hand, the leniency of u also affects the trust decision. u might be very lenient comparing to an overall user leniency giving higher ratings to the reviews and trusting the reviewers more often. The leniency is computed as the difference between the average rating that u gives to the reviews and the overall average rating in the dataset.

$$f_{7,b} = \check{r}_u - \bar{r}.$$

The last feature we include for this parameter is equal to the difference between the average rating given to v and the average rating given by u

$$f_{7,c} = \check{r}_v - \check{r}_u.$$



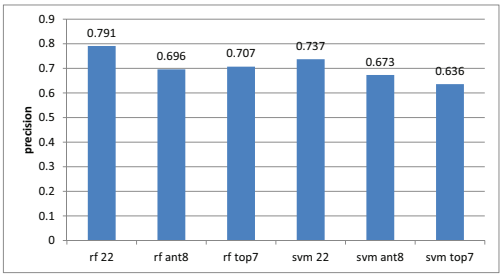


Fig. 1. Precision on 2,000, 000-million and 2,000-instance datasets, respectively

Parameter 8: *u and v have the same trustees.*

It is important to remember that computing the user similarity between a pair of users we are not necessarily constrained to the user rating information only. For example, this parameter captures the similarity between the sets of trustees of two users. The intuition behind it is that if two users have the same friends, they might be friends with each other as well.

$$f_8 = \frac{|T_u \cap T_v|}{|T_u \cup T_v|}.$$

4. Evaluation

4.1. Experimental Design

We compare our set of features with two other approaches. The first model denoted by *ant8* consists of eight features derived from the Antecedent Framework [4]. The second one, *top7*, includes top seven features from [3]. We denote our model using all 22 features by *rf22* (*rf* stands for Random Forests). Similarly, *rf7* consists of our top 7 features. The datasets generated for the experiments contain only trust and lack of trust statements, which allows our approach to be directly compared against the other methods. The 2-million dataset preserves the original distributions for trust and lack of trust statements. This provides a stratified experimental setup. Lastly, there exists review write-rate relationships between the trustor and trustee candidates in the dataset (i.e. the trustor gave a rating to one of the reviews produced by the trustee). This allows the Antecedent Framework model to score the candidate pairs from the data. The dataset generated for the first two experiments includes 400,000 trust statements and 1,600,000 randomly selected lack of trust statements.

We applied the Random Forests Classifier from Weka [7] with the number of trees equal to 30, and the

maximum depth of each tree equal to 100 in order to build the models for each set of features. The J48 algorithm is used to grow a single tree. The models were evaluated using a ten-fold cross-validation.[1] The top seven features of *top7* include features 1,2,4,5,6,8, and 9.
We also compare all three approaches using Random Forests and Support Vector Machines trained on the smaller dataset containing 1000 trusts and 1000 lack of trust statements. There is no comparison between *SVM* and *RF* on the 2-million dataset, as training the SVM classifier on the features suggested is inpracticable due to time constraints.

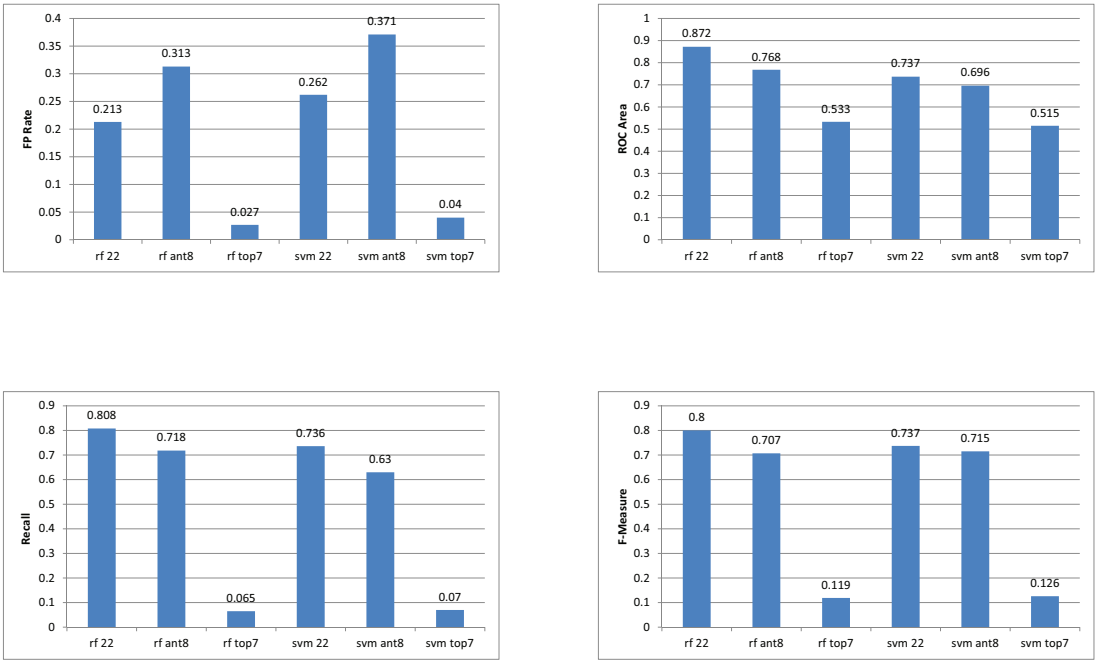


Fig. 2. Performance metrics for two classifiers on the 2,000-instance dataset

4.2. Random Forest and Support Vector Machine comparison

Figures 1 and 2 compare the results for the models constructed by Random Forest and Support Vector Machines from the 2000-instance dataset. In overall, RFs outperforms SVM on this dataset showing higher scores for precision, recall, F-measure and ROC Area. Using SVM reduces the scores for both models. *ant*8 and *top*7 appear to be a bit more stable than our model when using different classifiers. The scores for the two are only slightly worse than the ones received for our model, when using the SVM classifier. Our model gives the best results in precision among all models for both classifiers: 0.8 and 0.73. The recall scores for both classifiers are somewhat contradictory. Random Forest yields a better recall of 0.8 for our model, whereas SVM improves the recall for *svm_ant*8 up to 0.76 outperforming our model by 3%. SVM gives much tighter results for F-measure (0.737, 0.715, and 0.126) and ROC Area (0.737, 0.696, and 0.515) between all approaches, with our model performing better than the other two.

4.3. Random Forests models

The results of using the Random Forests classifier on the two-million instance dataset are summarized on Figures 1 and 3. In general, all metrics show higher scores for *rf22* and *rf7* over the other two models. *rf22* yields the best accuracy of 88.8% followed by *rf7* giving 87.3%. Our best result allows the accuracy to

[1] The features of *ant8* were computed with $\mu = 5$ and $\alpha = 0.1$.

be improved by 5% comparing to *ant8* (83.4%) and by nearly 9% comparing to *top7* (80.1%). *rf22* yields the best FP rate of 0.048 followed by *ant8* (0.056). *rf22* shows significant improvements of 5% and 20% in precision, over *ant8* (0.64) and *top7* (0.57) , respectively. The recall metrics for *rf22* and *rf7* is 20% greater than *ant8*. *top7* yields a recall rate of 0.029. F-measure reflects the precision and recall scores by showing a 4% improvement for our model (0.7) over *ant8* (0.66) Lastly, ROC Area shows that all classifiers perform better than random prediction. *rf22* and *rf7* show a 10% improvement over *ant8* for this metrics.

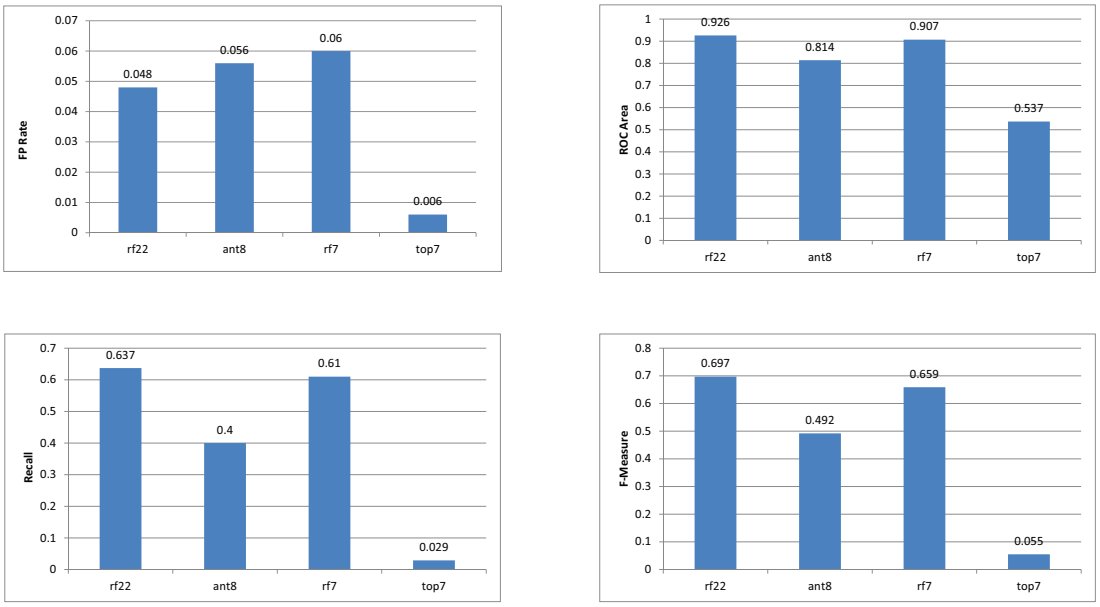


Fig. 3. Performance metrics for two classifiers on the 2,000,000-instance dataset

References

[1] J. A. Golbeck, Computing and applying trust in web-based social networks (2005).
[2] U. Kuter, J. Golbeck, Sunny: A new algorithm for trust inference in social networks using probabilistic confidence models., in: AAAI'07, 2007, pp. 1377–1382.
[3] H. Liu, E.-P. Lim, H. W. Lauw, M.-T. Le, A. Sun, J. Srivastava, Y. A. Kim, Predicting trusts among users of online communities: an epinions case study, in: Proceedings of the 9th ACM conference on Electronic commerce, EC '08, ACM, New York, NY, USA, 2008, pp. 310–319. doi:http://doi.acm.org/10.1145/1386790.1386838.
URL `http://doi.acm.org/10.1145/1386790.1386838`
[4] V.-A. Nguyen, E.-P. Lim, J. Jiang, A. Sun, To trust or not to trust? predicting online trusts using trust antecedent framework, in: Data Mining, 2009. ICDM '09. Ninth IEEE International Conference on, 2009, pp. 896 –901. doi:10.1109/ICDM.2009.115.
[5] N. Ma, E. peng Lim, H. Liu, Trust relationship prediction using online product review data.
[6] Y. Koren, Collaborative filtering with temporal dynamics, Commun. ACM 53 (2010) 89–97. doi:http://doi.acm.org/10.1145/1721654.1721677.
URL `http://doi.acm.org/10.1145/1721654.1721677`
[7] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten, The weka data mining software: an update, SIGKDD Explor. Newsl. 11 (1) (2009) 10–18. doi:10.1145/1656274.1656278.
URL `http://dx.doi.org/10.1145/1656274.1656278`